UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/801,951	03/08/2001	Philip G. Durr	206581	1464

23460 7590 02/10/2005

LEYDIG VOIT & MAYER, LTD
TWO PRUDENTIAL PLAZA, SUITE 4900
180 NORTH STETSON AVENUE
CHICAGO, IL 60601-6780

EXAMINER	
TANG, KUO LIANG J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 02/10/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/801,951	DURR ET AL.	
	Examiner	Art Unit	
	Kuo-Liang J Tang	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12/07/2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-20,22-38 and 40-45* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3-20,22-38 and 40-45* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This Office Action is in response to the amendment filed on 12/07/2004.

The priority date for this application is 3/8/2001.

Response to Arguments

2. Applicant's arguments, see REMARKS page 2, filed 12/07/2004, with respect to the Campbell reference (US Patent No. 6,513,133) is disqualified under 35 U.S.C. 103 (c) have been fully considered and are persuasive. The Final Rejection of Claims 1, 3-20, 22-28 and 40-45 has been withdrawn. However, upon further consideration, a new ground(s) of rejection is made in view of Ghosh.

Claims 1, 3-20, 22-38, 40-45 remain pending and have been examined. Claims 1, 12-14, 20, 31-32 and 38 are amended.

Claims 1, 6-12, 20, 25-30, 38 and 43-45 remain rejected under 35 U.S.C. 103(a) as being unpatentable over Baisley in view of Ghosh, further in view of Devanbu, further in view of Du.

Claims 3-5, 22-24 and 40-42 remain rejected under 35 U.S.C. 103(a) as being unpatentable over Baisley in view of Ghosh further in view of Richter.

Claims 13-19, 31-37 remain rejected under 35 U.S.C. 103(a) as being unpatentable over Richter, further in view of Ghosh, further in view of Du.

It should be noted that Admendment filed on 5/26/2004 also have been addressed and / or covered as set forth on this action.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1, 6-12, 20, 25-30, 38 and 43-45 are rejected under 35 U.S.C. 103(a) as being unpatentable over Baisley el al. US Patent No. 6,330,569 (hereinafter Baisley) in view of Ghosh US Patent No. 6,412,109, further in view of Devanbu, US Patent No. 5,832,271 (art of record), further in view of Du et al., US Patent No. 6,308,163 (hereinafter Du).

As Per Claim 1, Baisley teaches a computer system executing a repository program and having a memory, a method is disclosed for versioning a UML model in the repository in accordance with an updated XML representation of the UML model. The method includes the steps of identifying differences between UML objects in the UML model and XML objects in the XML file. Where there is a difference between one of the UML objects and one of the XML objects, it is reserved as a ghost object. (E.g. see Abstract and associated text). In that Baisley discloses the method that covering the steps of:

"receiving a set of original program segments referenced by the executable program;" (E.g. See col. 3:10-15, xml and UML objects);

"identifying a set of substitute program segments associated with ones of the original set of program segments;" (E.g. See FIG. 5A, blk 55, attributes);

"modifying the ones of the original set of program segments to include an exception inducing code;" (E.g. See Abstract); and

"creating an association between ones of the set of original program segments and ones of the set of substitute program segments." (Eg. See FIG. 6);

Baisley teaches "modifying the ones of the original set of program segments" (E.g. See FIG. 8B, blk 100). Baisley does not explicitly disclose including "an exception inducing code". However, Ghosh teaches in a manner such as modifying the ones of the original set of program segments to include an exception inducing code (E.g. See col. 4:47 to col. 6:67, which states "... int y=5/0; The assignment of a value to "y" will throw an exception because it involves the undefined division of an integer by zero. ... FIG. 3A is a flow diagram that demonstrates the disruption in the control flow of a program introduced by an exception. ...). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ghosh to induce a more uniform distribution of exceptions for all routines into Baisley's system. The modification would have been obvious because one of ordinary skill in the art would have been motivated so that the system is able to ensure proper exception handling by all routines.

The combination of Baisley and Ghosh does not explicitly disclose "executable binary code". However, Devanbu teaches in a manner such as modifying executable binary code of the ones of the original set of program segments to include an exception inducing code comprising executable binary code (E.g. See Devanbu col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Devanbu to include executable binary

code into the system of Baisley and and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to obtain information about dynamic properties of a program.

The combination of Baisley, Ghosh and Devanbu does not explicitly disclose "policy for substitution". However, Du teaches in a manner such as creating an association between ones of the set of original program segments and ones of the set of substitute program segments, each association comprising a policy for determining whether to execute the substitute program segment depending on an identity of a calling module of the original program segment .(E.g. See Du FIG. 5, policy engine 130, FIG. 6, step 162 and associated text, e.g. see col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Du to include policy for substitution into the system of Baisley, Ghosh and Devanbu. The modification would have been obvious because one of ordinary skill in the art would have been motivated to improve resource management in workflow processing of an enterprise.

As per Claim 6, the rejection of claim 1 is incorporated and further Baisley teaches

"building a table entry including a called original program segment reference (E.g. see FIG. 4, blk 36) and a substitute program segment reference. (E.g. see FIG. 4, blk 38) " (E.g. see col 3:15-20).

As per Claim 7, the rejection of claim 6 is incorporated and further Baisley teaches

"the called original program segment reference comprises an address." (E.g. see FIG. 4 and col. 3:47-50, which states "... map in memory ...").

As Per Claim 8, the rejection of claim 6 is incorporated and further Baisley teaches "storing a portion of the original program segment replaced by the code in table entry. (E.g. see Baisley col. 3:15-20") ". Baisley does not explicitly disclose "including an exception inducing code". However, Ghosh teaches in a manner such as modifying the ones of the original set of program segments to include an exception inducing code (E.g. See col. 5:54-67, fault induction code). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ghosh to induce a more uniform distribution of exceptions for all routines into Baisley's system. The modification would have been obvious because one of ordinary skill in the art would have been motivated so that the system is able to ensure proper exception handling by all routines.

As per Claim 9, the rejection of claim 6 is incorporated and further Baisley teaches

"storing a condition description specifying a program execution state (E.g. see FIG. 4, blk 47) under which a substitute program segment will not be executed in place of an associated called original program segment." (E.g. see col. 7:15-20, for not be executed, see "reserved").

As per Claim 10, the rejection of claim 1 is incorporated and further Baisley teaches

"searching (E.g. see FIG. 5A, step 53, traversal) a program modification database including a set of identified programs (E.g. see FIG. 4, blk 37, 41) and corresponding substitute program segments. (E.g. see FIG. 4, blk 36, 39)".

As per Claim 11, the rejection of claim 1 is incorporated and further Baisley teaches

"loading an executable program into active process space;" (E.g. see col. 8:19-35);

"identifying an in-memory patch within the program modification database corresponding to the executable program;" (E.g. see FIG. 4, blk 38, GHOST OBJECT ID) and

"inserting an in-memory patch within the executable program residing in the active process space." (E.g. see FIG. 4, blk 38, GHOST OBJECT ID; and col. 3:15-20).

As per Claim 12, Baisley teaches

"loading, for execution, a copy of the executable program in active process space;" (E.g. see col. 8:19-35);

"identifying a set of modifications corresponding to the executable program;" (E.g. see FIG. 4, blk 38, GHOST OBJECT ID) and

"modifying the copy of the executable program in active process space to include an exception inducing code, wherein the exception inducing code initiates execution of a handling routine for determining and executing at least a corresponding one of the set of modifications."

Baisley teaches "modifying the the executable program in active process space" (E.g. See FIG. 8B, blk 100). Baisley does not explicitly disclose including "an exception inducing code", wherein the exception inducing code initiates execution of a handling routine for determining and executing at least a corresponding one of the set of modifications. However, Ghosh teaches in a manner such as modifying the ones of the original set of program segments to include an exception inducing code, wherein the exception inducing code initiates execution of a handling routine for determining and executing at least a corresponding one of the set of modifications (E.g. See col. 5:54-67, fault induction code). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ghosh to induce a more uniform distribution of exceptions for all routines into Baisley's system. The modification would have been obvious because one of ordinary skill in the art would have been motivated so that the system is able to ensure proper exception handling by all routines.

The combination of Baisley and Ghosh does not explicitly disclose "executable binary code". However, Devanbu teaches in a manner such as modifying executable binary code of the ones of the original set of program segments to include an exception inducing code comprising executable binary code (E.g. See Devanbu col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the

invention was made to incorporate the teaching of Devanbu to include executable binary code into the system of Baisley and and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to obtain information about dynamic properties of a program.

The combination of Baisley, Ghosh and Devanbu does not explicitly disclose "policy for substitution". However, Du teaches in a manner such as creating an association between ones of the set of original program segments and ones of the set of substitute program segments, each association comprising a policy for determining whether to execute the substitute program segment depending on an identity of a calling module of the original program segment. (E.g. See Du FIG. 5, policy engine 130, FIG. 6, step 162 and associated text, e.g. see col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Du to include policy for substitution into the system of Baisley, Ghosh and Devanbu. The modification would have been obvious because one of ordinary skill in the art would have been motivated to improve resource management in workflow processing of an enterprise.

As per Claim 20, is the computer-readable medium claim corresponding to the method claim 1 and is rejected under the same reason set forth in connection of the rejection of claim 1. Further Baisley discloses machine-readable storage medium. (E.g. see col. 8:19-35).

As per Claims 25 and 29-30, the rejection of claim 20 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 6 and 10-11 respectfully.

As per Claims 26-28, the rejection of claim 25 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 7-8 respectfully.

As per Claim 38, Baisley teaches

"a program modification database for storing a set of substitute program segments associated with ones of the original set of program segments;" (E.g. see FIG. 2, blk 15, REPOSITORY);

"a program loader for installing a copy of the executable program into an active process space;" (E.g. see FIG. 2, blk 20, and col. 5:27-42);

"a substitution list (E.g. see FIG. 4, blk 35, 39) including entries describing an association between ones of the set of original program segments (E.g. see FIG. 4, blk 37, 41) and ones of the set of substitute program segments. (E.g. see FIG. 4, blk 38, 42)"

Baisley teaches "a program code modifier for altering, within the active process space, the ones of the original set of program segments" (E.g. See FIG. 8B, blk 100). Baisley does not explicitly disclose "including an exception inducing code". However, Ghosh teaches in a manner such as modifying the ones of the original set of program segments to include an exception inducing code (E.g. See col. 5:54-67, fault induction code). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ghosh to induce a more uniform distribution of exceptions for all routines into Baisley's system. The

modification would have been obvious because one of ordinary skill in the art would have been motivated so that the system is able to ensure proper exception handling by all routines.

The combination of Baisley and Ghosh does not explicitly disclose "policy for substitution". However, Du teaches in a manner such as creating an association between ones of the set of original program segments and ones of the set of substitute program segments, each association comprising a policy for determining whether to execute the substitute program segment depending on an identity of a calling module of the original program segment. (E.g. See Du FIG. 5, policy engine 130, FIG. 6, step 162 and associated text, e.g. see col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Du to include policy for substitution into the system of Baisley and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to improve resource management in workflow processing of an enterprise.

As per Claim 43, the rejection of claim 38 is incorporated and further Baisley teaches

"individual entries within the substitution list included a called original program segment reference (E.g. see FIG. 4, blk 37, 41) and a substitute program segment reference. (E.g. see FIG. 4, blk 38, 42)".

As per Claim 44, the rejection of claim 43 is incorporated and is rejected under the same reason set forth in connection of the rejection of claim 8.

As per Claim 45, the rejection of claim 38 is incorporated and further Baisley teaches

"an in-memory patch within the program modification database corresponding to the executable program;" (E.g. see FIG. 4, blk 38, GHOST OBJECT ID) and

"an in-memory patching function for inserting the in-memory patch within the executable program residing in the active process space." (E.g. see FIG. 4, blk 38, GHOST OBJECT ID; and col. 3:15-20).

4. Claims 3-5, 22-24 and 40-42 are rejected under 35 U.S.C. 103(a) as being unpatentable over Baisley in view of Ghosh further in view of Jeffrey Richter, "Programming Applications Microsoft Windows" Fourth Edition, pages 794-800 (art of record, hereinafter Richter).

As Per Claim 3, the rejection of claim 1 is incorporated and further Baisley and Ghosh teach program interface (E.g. see Baisley, FIG.2 blk 20). Baisley and Ghosh do not explicitly disclose "the substitute program segments comprise functions". However, Richter teaches in a manner such as the substitute program segments comprise functions (E.g. See page 795, last line, company's ExitProcess replacement function.) and (E.g. see page 796, API Hooking by Overwriting Code, step 1). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to

incorporate the teaching of Richter to induce a more uniform distribution of exceptions for all routines into system of Baisley and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to use company's ExitProcess replacement function so that the company's DLLs perform all of its cleanup successfully and then call the operating system ExitProcess to clean up all DLLs.

As Per Claim 4, the rejection of claim 3 is incorporated and further Baisley and Ghosh teach program interface (E.g. see Baisley, FIG.2 blk 20). Baisley and Ghosh do not explicitly disclose "the functions comprise application program interfaces". However, Richter teaches in a manner such as the functions comprise application program interfaces (E.g. See page 795, last line, company's ExitProcess replacement function.) and (E.g. see page 796, API Hooking by Overwriting Code, see API). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Richter to induce a more uniform distribution of exceptions for all routines into system of Baisley and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to use company's ExitProcess replacement function so that the company's DLLs perform all of its cleanup successfully and then call the operating system ExitProcess to clean up all DLLs.

As Per Claim 5, the rejection of claim 4 is incorporated and further Baisley and Ghosh teach program interface (E.g. see Baisley, FIG.2 blk 20). Baisley and Ghosh do

not explicitly disclose "the application program interfaces are part of an operating system". However, Richter teaches in a manner such as the application program interfaces are part of an operating system (E.g. See page 795, 3rd paragraph, operating system's ExitProcess function.) and (E.g. see page 796, API Hooking by Overwriting Code, step 1, kernel32.dll). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Richter to induce a more uniform distribution of exceptions for all routines into system of Baisley and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to use company's ExitProcess replacement function so that the company's DLLs perform all of its cleanup successfully and then call the operating system ExitProcess to clean up all DLLs.

As per Claims 22-24, the rejection of claim 20 is incorporated and and are rejected under the same reason set forth in connection of the rejection of claims 3-5 respectfully.

As per Claims 40-42, the rejection of claim 38 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 3-5 respectfully.

5. Claims 13-19, 31-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Richter, further in view of Ghosh, further in view of Du et al., US Patent No. 6,308,163 (hereinafter Du).

As Per Claim 13, Richter teaches:

"detecting the exception inducing code;" (E.g. see Richter page 796, API Hooking by Overwriting Code, steps 1-7);

"determining a corresponding substitute program segment entry in a program segment substitution list;" (E.g. see Richter page 796, API Hooking by Overwriting Code, step 1); and

"executing the substitute program segment in place of the original program segment." (E.g. see Richter page 796, API Hooking by Overwriting Code, step 4).

Richter teaches "receiving a set of original program segments referenced by the executable program;" (E.g. see Richter page 795, last paragraph, company's DLL modules and ExitProcess function). Richter does not explicitly disclose "including an exception inducing code", wherein the exception inducing code initiates execution of a handling routine for determining and executing at least a corresponding one of the set of modifications. However, Ghosh teaches in a manner such as calling from a calling module having an identity, an original program segment of the active executable program that has been modified to include an exception inducing code, the exception inducing code comprising executable binary code that induces an exception handled by an exception handling routine (E.g. See col. 5:54-67, fault induction code). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Ghosh to induce a more uniform distribution of exceptions for all routines into Richter's system. The modification would have been obvious because one of ordinary skill in the art would have been motivated so that the system is able to ensure proper exception handling by all routines.

The combination of Richter and Ghosh does not explicitly disclose "policy for substitution". However, Du teaches in a manner such as determining whether to execute the substitute program segment depending on an identity of a calling module (E.g. See Du FIG. 5, policy engine 130, FIG. 6, step 162 and associated text, e.g. see col. 4:49-57). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Du to include policy for substitution into the system of Richter and Ghosh. The modification would have been obvious because one of ordinary skill in the art would have been motivated to improve resource management in workflow processing of an enterprise.

As per Claim 14, the rejection of claim 13 is incorporated and further Richter teaches

"determining, by reference to an exclusion/inclusion policy associated with the instance of the exception inducing code, whether to perform the substitute program segment in place of the original program segment, wherein the exclusion/inclusion policy identifies at least a condition depending on the identity of the calling module under which the corresponding substitute program segment will not be executed in place of an associated original program segment." (E.g. see page 797, API Hooking by Manipulating a Modules's Import Section and page 798, lines 7-8 for exclusion/not be executed).

As per Claim 15, the rejection of claim 14 is incorporated and further Richter teaches

"the corresponding substitute program segment entry includes a portion of the original program segment replaced by the exception Inducing code," (E.g. see page 797, API Hooking by Manipulating a Modules's Import Section) and

"further comprising the step of determining that the condition under which the corresponding substitute program segment will not be executed has been fulfilled, and in response restoring (E.g. see page 798, line 8, return)within a program code execution stream the portion of the original program segment replaced by the exception inducing code." (E.g. see page 797, API Hooking by Manipulating a Modules's Import Section and page 798, lines 7-8 for exclusion/not be executed).

As per Claim 16, the rejection of claim 13 is incorporated and further Richter teaches

"the substitute program segments comprise functions." (E.g. see page 796, API Hooking by Overwriting Code, step 4).

As per Claim 17, the rejection of claim 16 is incorporated and further Richter teaches

"the functions comprise application program Interfaces." (E.g. see page 796, API Hooking by Overwriting Code, see API).

As per Claim 18, the rejection of claim 17 is incorporated and further Richter teaches

"the application program interfaces are part of an operating system." (E.g. see page 796, API Hooking by Overwriting Code, step 1, Kernel32.dll).

As per Claim 19, the rejection of claim 13 is incorporated and further Richter teaches

"the corresponding substitute program segment entry includes a called original program segment reference and a substitute program segment reference." (E.g. see page 796, API Hooking by Overwriting Code, step 3).

As per Claim 31, is the computer-readable medium claim corresponding to the method claim 13 and is rejected under the same reason set forth in connection of the rejection of claim 13.

As per Claims 32-33 and 37, the rejection of claim 31 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 14-15 and 19 respectfully.

As per Claims 34-36, the rejection of claim 32 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 16-18 respectfully.

Conclusion

6. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Correspondence Information

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kuo-Liang J Tang whose telephone number is 703-305-4866. The examiner can normally be reached on 8:30AM - 5:00PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 703-305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

After October 25, 2004, examiner can be reached at new telephone number (571) 272-3705, and the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Kuo-Liang J. Tang

Software Engineer Patent Examiner

WEI Y. ZHEN
PRIMARY EXAMINER